

ACM321 Javalab 12 GUI

Java Swing

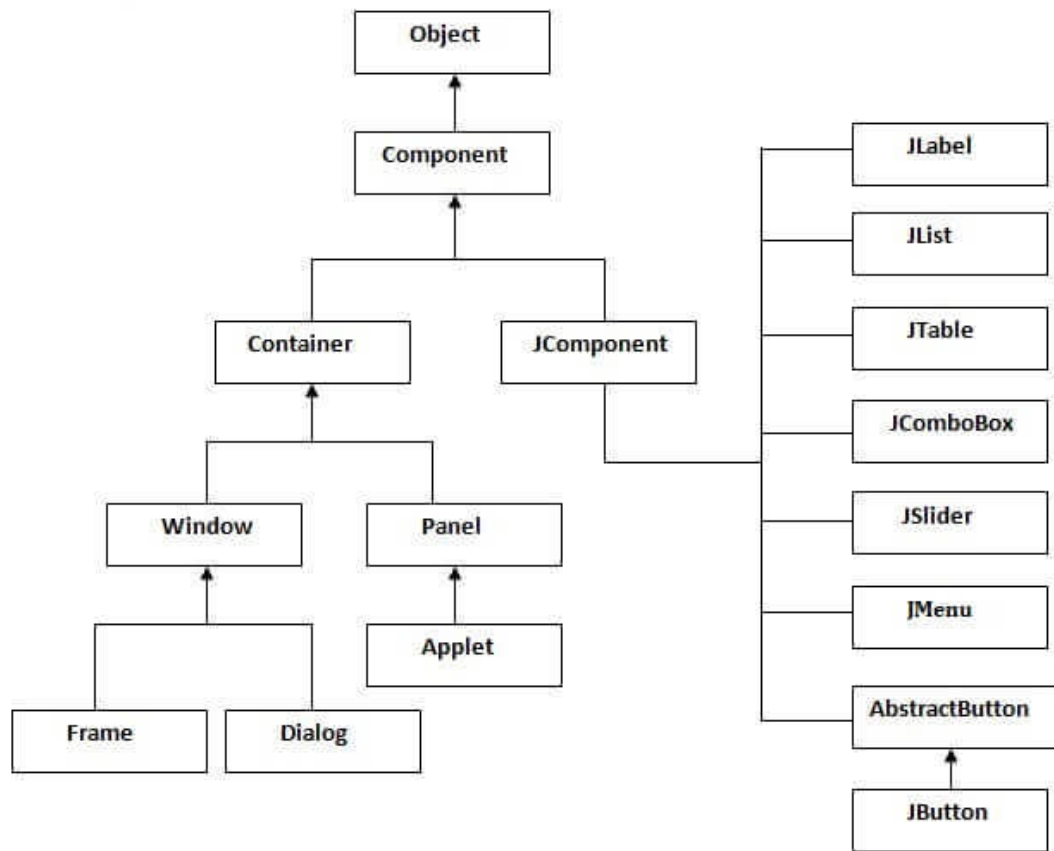
Java Swing is a lightweight Java graphical user interface (GUI) widget toolkit that includes a rich set of widgets. Swing includes built-in controls such as trees, image buttons, tabbed panes, sliders, toolbars, color choosers, tables, and text areas to display HTTP or rich text format (RTF). Swing components are written entirely in Java and thus are platform-independent.

Swing offers customization of the look and feel of every component in an application without making significant changes to the application code. It also includes a pluggable look and feel feature, which allows it to emulate the appearance of native components while still having the advantage of platform independence. This particular feature makes writing applications in Swing easy and distinguishes it from other native programs.

Hierarchy of Java Swing Classes

Hierarchy of Java Swing classes

The hierarchy of java swing API is given below.

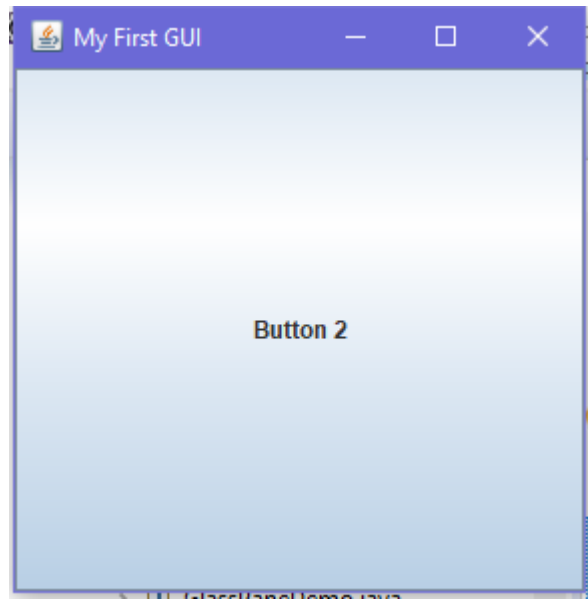


What is a container class?

Container classes are classes that can have other components on it. So for creating a GUI, we need at least one container object. There are 3 types of containers.

1. **Panel:** It is a pure container and is not a window in itself. The sole purpose of a Panel is to organize the components on to a window.
2. **Frame:** It is a fully functioning window with its title and icons.
3. **Dialog:** It can be thought of like a pop-up window that pops out when a message has to be displayed. It is not a fully functioning window like the Frame.

Java GUI Example



```
package javalab_gui;
import javax.swing.*;

//Usually you will require both swing and awt packages
//even if you are working with just swings.
import java.awt.*;

public class ProgramMain {
    public static void main(String[] args) {
        //frame
        JFrame frame = new JFrame("My First GUI");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(300,300);

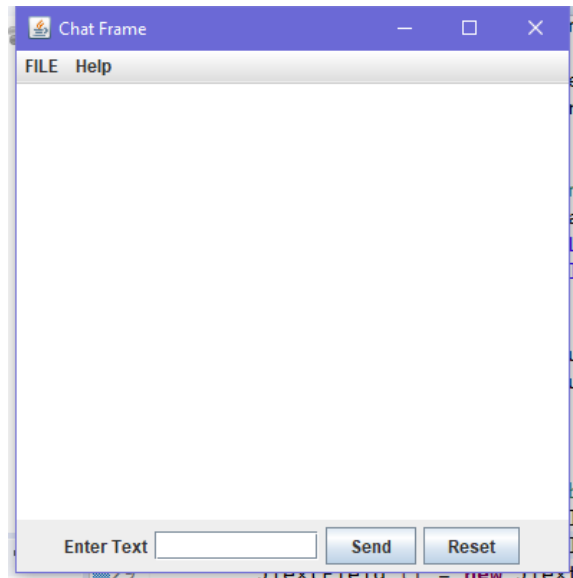
        //button
        JButton button = new JButton("Press");
        frame.getContentPane().add(button); // Adds Button to content pane of frame

        //add second button
        JButton button2 = new JButton("Button 2");
        frame.getContentPane().add(button2);

        //set frame visible
        frame.setVisible(true);

        //we need a layout manager to get rid of the overlapping problem

    }
}
```



```
package javalab_gui;

import javax.swing.*;

//Usually you will require both swing and awt packages
//even if you are working with just swings.
import java.awt.*;

public class ProgramMain {
    public static void main(String[] args) {
        //Creating the Frame
        JFrame frame = new JFrame("Chat Frame");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 400);

        //Creating the MenuBar and adding components
        JMenuBar mb = new JMenuBar();
        JMenu m1 = new JMenu("FILE");
        JMenu m2 = new JMenu("Help");
        mb.add(m1);
        mb.add(m2);
        JMenuItem m11 = new JMenuItem("Open");
        JMenuItem m22 = new JMenuItem("Save as");
        m1.add(m11);
        m1.add(m22);

        //Creating the panel at bottom and adding components
        JPanel panel = new JPanel(); // the panel is not visible in output
        JLabel label = new JLabel("Enter Text");
        JTextField tf = new JTextField(10); // accepts upto 10 characters
        JButton send = new JButton("Send");
        JButton reset = new JButton("Reset");
        panel.add(label); // Components Added using Flow Layout
        panel.add(tf);
```

```

        panel.add(send);
        panel.add(reset);

        // Text Area at the Center
        JTextArea ta = new JTextArea();

        //Adding Components to the frame.
        frame.getContentPane().add(BorderLayout.SOUTH, panel);
        frame.getContentPane().add(BorderLayout.NORTH, mb);
        frame.getContentPane().add(BorderLayout.CENTER, ta);
        frame.setVisible(true);
    }
}

```

ComboBox Demo

```

package components;

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

/*
 * ComboBoxDemo.java uses these additional files:
 *   images/Bird.gif
 *   images/Cat.gif
 *   images/Dog.gif
 *   images/Rabbit.gif
 *   images/Pig.gif
 */
public class ComboBoxDemo extends JPanel
    implements ActionListener {

    JLabel picture;

    public ComboBoxDemo() {
        super(new BorderLayout());

        String[] petStrings = { "Bird", "Cat", "Dog", "Rabbit", "Pig" };

        //Create the combo box, select the item at index 4.
        //Indices start at 0, so 4 specifies the pig.
        JComboBox petList = new JComboBox(petStrings);
        petList.setSelectedIndex(4);
        petList.addActionListener(this);

        //Set up the picture.
        picture = new JLabel();
        picture.setFont(picture.getFont().deriveFont(Font.ITALIC));
        picture.setHorizontalAlignment(JLabel.CENTER);
        updateLabel(petStrings[petList.getSelectedIndex()]);
        picture.setBorder(BorderFactory.createEmptyBorder(10,0,0,0));
    }

    public void actionPerformed(ActionEvent e) {
        updateLabel(petStrings[petList.getSelectedIndex()]);
    }

    private void updateLabel(String petString) {
        picture.setText(petString);
    }
}

```

```

        //The preferred size is hard-coded to be the width of the
        //widest image and the height of the tallest image + the border.
        //A real program would compute this.
        picture.setPreferredSize(new Dimension(177, 122+10));

        //Lay out the demo.
        add(petList, BorderLayout.PAGE_START);
        add(picture, BorderLayout.PAGE_END);
        setBorder(BorderFactory.createEmptyBorder(20,20,20,20));
    }

    /** Listens to the combo box. */
    public void actionPerformed(ActionEvent e) {
        JComboBox cb = (JComboBox)e.getSource();
        String petName = (String)cb.getSelectedItem();
        updateLabel(petName);
    }

    protected void updateLabel(String name) {
        ImageIcon icon = createImageIcon("images/" + name + ".gif");
        picture.setIcon(icon);
        picture.setToolTipText("A drawing of a " + name.toLowerCase());
        if (icon != null) {
            picture.setText(null);
        } else {
            picture.setText("Image not found");
        }
    }

    /** Returns an ImageIcon, or null if the path was invalid. */
    protected static ImageIcon createImageIcon(String path) {
        java.net.URL imgURL = ComboBoxDemo.class.getResource(path);
        if (imgURL != null) {
            return new ImageIcon(imgURL);
        } else {
            System.err.println("Couldn't find file: " + path);
            return null;
        }
    }

    /**
     * Create the GUI and show it. For thread safety,
     * this method should be invoked from the
     * event-dispatching thread.
     */
    private static void createAndShowGUI() {
        //Create and set up the window.
        JFrame frame = new JFrame("ComboBoxDemo");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        //Create and set up the content pane.
        JComponent newContentPane = new ComboBoxDemo();
        newContentPane.setOpaque(true); //content panes must be opaque
        frame.setContentPane(newContentPane);

        //Display the window.
        frame.pack();
        frame.setVisible(true);
    }

```

```


    }

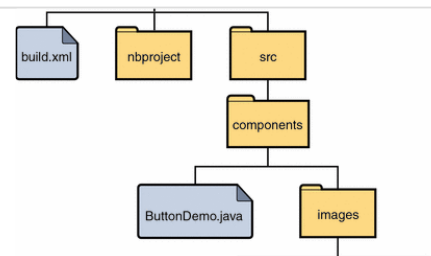
    public static void main(String[] args) {
        //Schedule a job for the event-dispatching thread:
        //creating and showing this application's GUI.
        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }
}

```

Using Swing Components: Examples

The table that follows lists every example in the Using Swing Components lesson, with links to required files and to where each example is discussed. The first column of the table has

 <https://docs.oracle.com/javase/tutorial/uiswing/examples/components/index.html>



What is Java Swing? - Definition from Techopedia

Java Swing Definition - Java Swing is a lightweight Java graphical user interface (GUI) widget toolkit that includes a rich set of widgets. It is part...

 <https://www.techopedia.com/definition/26102/java-swing>

